5-1-2015

mCollector: Sensor-enabled health-data collection system for rural areas in the developing world

Rima Narayana Murthy
Dartmouth College

mCollector: Sensor-enabled health-data collection system for rural areas in the developing world

A Thesis

Submitted to the Faculty

in partial fulfillment of the requirements for the

degree of

Master

of

Science

by

Rima Narayana Murthy

DARTMOUTH COLLEGE

Hanover, New Hampshire

May 2015

Examining Committee:

(Advisor) David F. Kotz, Ph.D.

Andrew Campbell, Ph.D.

Guanling Chen, Ph.D

Dartmouth Computer Science Technical Report TR2015-788

Abstract

Health data collection poses unique challenges in rural areas of the developing world. mHealth systems that are used by health workers to collect data in remote rural regions should also record contextual information to increase confidence in the fidelity of the collected data.

We built a user-friendly, mobile health-data collection system using wireless medical sensors that interface with an Android application. The data-collection system was designed to support minimally trained, non-clinical health workers to gather data about blood pressure and body weight using off-the-shelf medical sensors. This system comprises a blood-pressure cuff, a weighing scale and a portable point-of-sales printer. With this system, we introduced a new method to record contextual information associated with a blood-pressure reading using a tablet's touchscreen and accelerometer. This contextual information can be used to verify that a patient's lower arm remained well-supported and stationary during her blood-pressure measurement. This method can allow mHealth applications to guide untrained patients (or health workers) in measuring blood pressure correctly.

Usability is a particularly important design and deployment challenge in remote, rural areas, given the limited resources for technology training and support. We conducted a field study to assess our system's usability in rural India, where we logged health worker interactions with the app's interface using an existing usability toolkit. Researchers analyzed logs from this toolkit to evaluate the app's user experience and quantify specific usability challenges in the app. We have recorded experiential notes from the field study in this document.

We present four contributions to future mHealth projects in this document:

- We describe a method for measuring lower-arm stillness and support during a blood-pressure measurement, using an off-the-shelf Android tablet.
- We evaluate our method for measuring lower-arm stillness with a preliminary

user study of 12 subjects and found that our method can distinguish stationary arms from different types of lower-arm movement with 90% accuracy.

- We conduct an experiential study with 28 participants and three app operators. In this study, we evaluate our system's field usability by deploying it in rural India.
- We provide a quantitative usability analysis of our mobile-data-collection app's interface using an existing usability toolkit.

Contents

List of Figures

Acknowledgments

This research results from a research program at the Institute for Security, Technology, and Society at Dartmouth College, supported by the National Science Foundation under award numbers 1016823 and 1143548. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation. First and foremost, I remain grateful for my advisor's guidance, patience and kindness that have given my work and my graduate student life purpose and meaning. He has been what every mentor should aspire to be.

A note of thanks to Aarathi Prasad (doctoral candidate at Dartmouth College) for introducing me to the challenges in collecting contextual information of mHealth data and immediate applications of the same through her research. Thank you, Aarathi, for the cupcakes, *jalebis* and unwavering good cheer you brought to lab 241. I thank Shrirang Mare (doctoral candidate) and Dr. Ashok Chandrashekar, both from Dartmouth College, for their support in the formulation of the methods presented here. Shrirang also conducted a difficult, rural usability study on my behalf at very short notice, thank you for that. Both the mCollector app and the method for classification followed Prof.Campbell's Smartphone sensing course I took in the winter of 2012, thank you for the great introduction! Associate Professor Guanling Chen and Dr. Xiaoxiao Ma from University of Massachusetts, Lowell have collaborated with us closely on this project and I thank them sincerely for their cooperation and support. Many thanks to my defense committee members, Professor Campbell and Associate Professor Chen, for their insightful feedback and guidance throughout this project. I also thank the anonymous reviewers at the NetHealth 2014 workshop for their feedback.

A personal note of thanks to my family for always being there and supporting my

work in every way they could and did: my father and my husband Ashok, for encouraging me to pursue higher education, my mother for believing that I could get where I am with all her heart and making me believe it too, and, my in-laws for all their help extended during my rural user study. Thank you, Ashok, for all the evenings spent discussing my work, for your constant stream of wisdom that I have ignored to my own peril, for all the kind words spoken and unspoken, and, for showing me by example, to always choose the path of reason.

And finally, a big hug and thank you to my chubby little son Aniketh. I look forward to this most fantastic adventure of growing up all over again, with you.

1 Introduction

A survey report jointly sponsored by the United Nations and Vodafone Foundation predicts that the coming decades will see a fundamental shift in health concerns of the developing world; from late-stage detection and treatment of chronic diseases such as cardiovascular diseases, hypertension and diabetes to early-stage detection and prevention of the same [**?**]. The same study, supported by statistics from a World Health Organization report, estimates that diabetes, heart disease, and stroke together will cost India nearly USD 336 billion in the next decade.

As low-cost smartphones and tablets become commonplace in the developing world, mobile data collection apps are increasingly used for health services in the field and for public health monitoring [**?**, **?**]. Mobile apps that interface with personal medical sensors have proven particularly effective for community health-data collection [**?**]. Any mHealth data collection system should include contextual information, wherever pertinent, along with the medical readings to enhance fidelity of the health data, as proposed by Prasad et al. in their provenance framework for mHealth [**?**].

We present a method to monitor lower-arm support and movement during blood-pressure measurement using an Android tablet's touchscreen interface and built-in accelerometer and without the use of dedicated sensors for monitoring blood-pressure (BP) posture. We chose to work with the Android-based tablets because several mHealth data collection solutions have already been built on this open-source framework [**?**, **?**, **?**]. Furthermore, we expect that Android tablets will be deployed widely for health data collection since tablet sales have already surpassed personal computer sales in some developing countries [**?**]. As the density of low-cost, next-generation smartphones and tablets increase in the developing world, systems that are used to collect health data can benefit from apps that interface with off-the-shelf medical sensors designed for personal

health monitoring of consumers in developed countries. However, there are several technical challenges with the design and deployment of such health-data collection systems:

- *Ensuring mobility*

  An mHealth data collection system must be compact and lightweight. Devices used in the system should run on battery power because electrical supply cannot be guaranteed. Further, data storage and synchronization services must be provisioned for the mobile devices so that data collected during the screening can reside on the device until it can be transferred to a secure server in the cloud.

  With the advent of Bluetooth 4.0, low-energy sensors that track personal health metrics accurately are likely to become common, thus driving down the cost of such sensors. As these sensors permeate homes in the western world, they could also be pertinent inclusions to mHealth projects in the developing world, given their low energy requirements, portable size and seamless integration with mobile devices [**?**].

- *Usability*

  Since the system is more likely to be used by minimally trained, non-technical workers to collect health data, the interface should be simple and intuitive to minimize human errors during data entry. It has been shown in previous studies that data-entry error using electronic forms are more prevalent in rural areas of the developing world than urban areas, thus affecting overall data fidelity [**?**]. Ease of use must be verifiable to scale health-data collection systems to cover entire populations.

- *Contextual information or "Provenance"*

  When workers with limited clinical skills are required to collect health data, the

system should provide a mechanism to record pertinent context to verify that the medical data was recorded per clinical standards. This clinical context, which serves to increase confidence in the correctness of health data, is referred to as *provenance* in this document. Any mHealth data collection solution created for rural workers should include provenance information, wherever possible, to evaluate or enhance fidelity of the reading obtained by clinically unqualified workers.

- *Security and privacy*

  Health information of individual participants could be sensitive for socio-cultural reasons; some health parameters may have stigma attached in certain cultures. This could cause subjects to be wary or uncomfortable with sharing their health information with health workers. Further, local laws may restrict or prevent organizations from maintaining databases containing identifiable information with corresponding health data. Therefore, a secure mHealth interface should:

  - minimize verbal query/response and manual data entry to input health data,
  - encrypt data collected on mobile devices,
  - share identifiable information only on a need-to-know basis, for example, with direct healthcare providers, and
  - support a de-identified, aggregated view of collected health data to share with public health officials.

- *Language localization*

  The system must support language localization to easily adapt to workers from diverse backgrounds and to enable literate but non-technical workers to use the system correctly.

In this thesis, we present an mHealth data collection system that includes contextual information associated with a blood-pressure measurement. Research has indicated that health workers are more prone to error in blood-pressure measurement than certified nurses, who are themselves more prone to error than physicians [**?**, **?**]. Therefore, it is useful to integrate a correcting component with mHealth data-collection systems to guide non-clinical workers in the field while they measure a patient's blood pressure and to record contextual data about how the blood-pressure was recorded in the field. We developed a mobile app, referred to as *mCollector* throughout the rest of this document, to study factors that affect usability in the field. This app interfaces with off-the-shelf, Bluetooth-enabled medical sensors. We chose sensors designed for personal use to ensure that non-clinical workers can easily operate these sensors to collect health data. We refer to these workers as *operators* in this document. The mCollector is given to operators who may have received little or no previous training to measure medical metrics such as blood pressure. The mCollector has a series of screens collecting data about a participant's medical history, blood pressure and body weight; at the end it prints a health report of the collected data. On the screens that are configured to display data collected from the medical sensors, the app polls the sensor to check whether a new reading is available. If a reading is available on the sensor device's memory, the app updates the screen with the new reading without requiring the operator to enter the medical reading manually. Additionally, to ensure that a clinically unskilled operator measured a patient's blood pressure correctly, we introduce a new method to record lower-arm position using the tablet's touchscreen interface. Data collected via the app's interface is encrypted and stored on the tablet, and synchronized to the cloud when the network is available.

We make four contributions to future mHealth projects:

- We describe a method for measuring lower-arm stillness and support during a

blood-pressure measurement, using an off-the-shelf Android tablet.

- We evaluate our method for measuring lower-arm stillness with a preliminary user study of 12 subjects and found that our method can distinguish stationary arms from different types of lower-arm movement with 90% accuracy.

- We conduct an experiential study with 28 participants and three app operators. In this study, we evaluate our system's field usability by deploying it in rural India.

- We provide a quantitative usability analysis of our mobile-data-collection app's interface using an existing usability toolkit.

We built a mobile-health data-collection system by integrating wireless medical sensors with an Android app. The app interface was designed for typical rural health workers who are less technologically aware than their peers in developed nations and possess no formal clinical training. Our goal was to study experiential challenges that may arise in rural deployments of mobile-health data-collection systems and to verify whether a subset of these challenges can be met by using fully configured sensors that support automatic data transfer, thereby minimizing manual data entry of clinical readings.

This app was designed to be used by health workers to collect blood pressure and body mass index (BMI). These health metrics are typically collected in health screenings and are also monitored periodically for cardiac patients under rehabilitation as they are good indicators of heart health [**?**]. We introduce a new method to guide health workers to maintain a recommended lower-arm posture of participants while measuring blood pressure, without the use of additional sensors. The multi-touch touchscreen interface and accelerometer readings are used to ensure that:

1. the blood-pressure sensor is activated only when the left arm is well-rested, and

Figure 1: Screening camp showing booths for haemoglobin (left) and height (right) measurements.

2. the lower arm remains stationary for the duration of the blood-pressure measurement by the sensor.

Contextual information, namely, tablet orientation and arm position, are recorded along with the corresponding blood-pressure reading.

We conducted a pilot study mCollector at screening centers in India, specifically, at cardiac screening camps organized by Narayana Hrudayalaya Hospitals (NHH) in Bangalore that cater to remote, rural villages in the state of Karnataka. The mobile screening camps we visited in India were pipelined with separate booths for participant registration, body height, body weight, blood pressure, ECG and so on as shown in Figure 1. Each booth is manned by different health workers, thus requiring a participant to carry her paper form from one booth to another for updating. In our study, we set up one booth at the screening camp. Our booth was manned by a non-clinical worker equipped with the mCollector. The mCollector used two off-the-shelf medical sensors, a blood-pressure cuff and a weighing scale. The mCollector app is a static series of screens, designed in a format similar to paper forms that are typically used in health screenings.

We introduce a new method to enable non-clinical workers to measure blood pressure correctly and without the use of additional external sensors. The touchscreen interface

of the tablet computer along with audio feedback is used to guide health workers to measure blood pressure per clinical standards. The mCollectorApp source code has been instrumented with an existing toolkit developed by collaborating researchers at the University of Massachusetts, Lowell (UMass) [**?**]. We refer to this simply as the *toolkit* throughout the rest of this document. It logs fine-grained user-interface (UI) events such as taps, swipes and button clicks. UI events generated by an "expert" user (or the app developer) are used to generate a *reference usage model* that describes correct or expected interface usage. The UI events generated by workers in the field are later compared with the reference model. The differences between the expert and the operator usage models can be used to discover and fix potential problems in interface design.

Data that is collected at the screening event is stored on the Android tablet and secured using hybrid encryption. This data is then transferred to a server in the cloud when the tablet enters a secured, wireless network. Encrypted data that resides on the tablet includes the participant's profile (index, age, gender and health metrics), MAC addresses of the sensors, provenance data, location co-ordinates and screening timestamp.

All sensors in the system use Bluetooth for data transfer and are pre-paired with the tablet. While this enhances overall system usability, it ties a set of sensors with a tablet. The cost of this configuration is acceptable because it enables operators to remain agnostic to sensor discovery and setup.

2 Related work

Recent research has focused on mobile technology as the candidate technology for improving health-data collection in remote or isolated regions of the developing world [**?**, **?**]. Surveys of the rural mHealth solutions landscape yield many successful, open-source projects that lower technical barriers by providing simpler interfaces to create survey

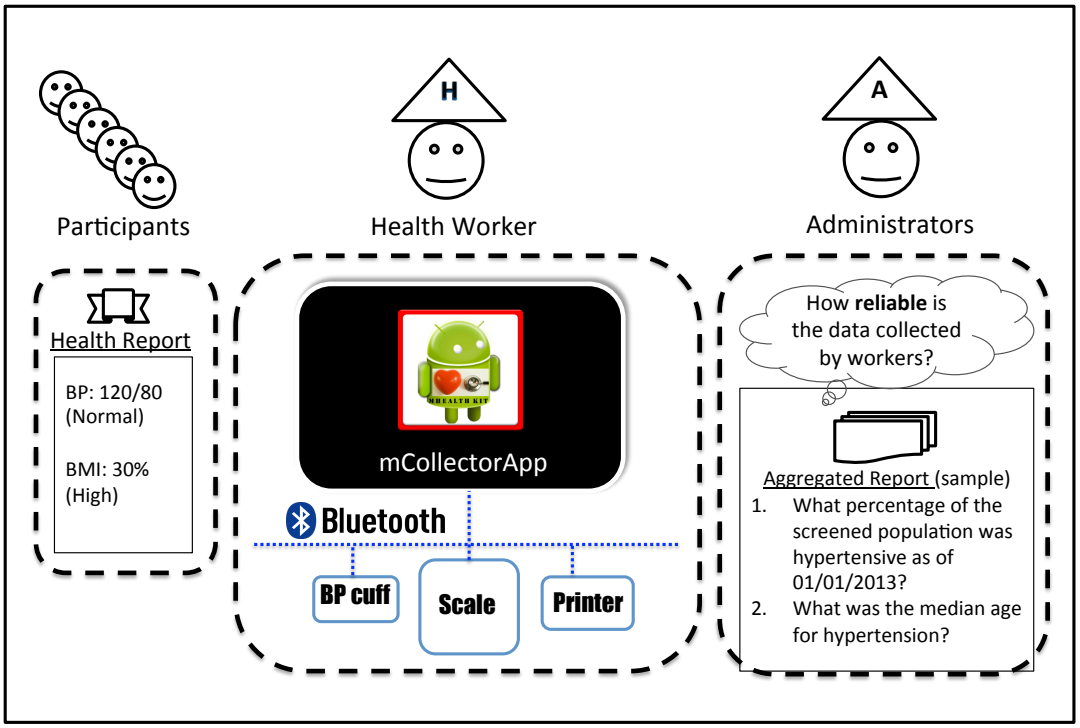

Figure 2: Mobile health-data collection architecture

forms, mobile apps to collect data, and integration with servers in the cloud to analyze data collected from the field in real-time [**?**]. As an example, consider Magpi, formerly known as Episurveyor, one of the first population-scale mHealth deployments handed out PDAs to Ugandan health workers to collect disease surveillance data directly from the field and share it in real-time with public health officials and other workers [**?**].

Mobile health (mHealth) solutions that use existing cellular networking infrastructure and information communication technologies are well-suited to support remote health monitoring and manage delivery of preventive public healthcare services. For example, SMS (Short Messaging Service) alerts have proven useful in raising public health awareness of communicable diseases such as HIV in Africa; similar alerts have also been

effective in remotely monitoring patient adherence to medication in clinical diabetes management programs [**?**].

During our survey of mobile data collection frameworks [**?**, **?**, **?**, **?**], we encountered many projects that address a wide range of medical-data-related challenges. These projects focused on raising health risks awareness, enabling user education and training, supporting epidemiological surveys that manage contagions, routing medical supplies to areas in critical need and remotely delivering medical care using mobile-powered telemedicine. The non-exhaustive list of pertinent projects summarized in this section introduces current state-of-the-art in rural mHealth data collection.

Magpi is a low-cost mobile health-data collection offering from DataDyne Group LLC that has been widely used across the African subcontinent for various epidemiological studies [**?**]. The Ugandan health information network (UHIN) was one such remote data collection program that employed health operators to use low-cost PDAs to collect public health-data at the community level. The collected data was then uploaded to a server at a rural health facility over infrared, Bluetooth, or Wi-Fi. However, it was discovered that the high cost of fuel prevented health officials from going out to collect data. Further, the short battery life of some PDAs caused loss of data [**?**]. Both of these issues significantly reduced the quantity and quality of data available for decision making. Although the latest version of Magpi (released in January 2013) supports mobile phones with better batteries than PDAs and provides GPS location tracking on its platform, there is no support for integrating external sensors.

The *sana* framework, developed at MIT, is a compelling health-data collection solution that integrates with OpenMRS, an open-source electronic medical record system platform [**?**]. Sana apps installed on Android devices can be used by health operators in the field to collect data, which is then uploaded to an OpenMRS platform. Authenticated doctors can then access these records remotely. This platform focuses on doctors viewing

medical data collected by nurses and trained clinicians, unlike our scenario where health data can be collected by non-clinical volunteers using external sensors. Further, sana does not keep data encrypted on the inherently insecure mobile devices, relying instead on the underlying Android platform to provide hardware encryption for those developers interested in encryption.

Open Data Kit (ODK) 1.0 is an open-source framework developed at the University of Washington that integrates different components for building survey forms, mobile data collection and visualization on the web [**?**]. ODK is also built on Google's Android operating system and uses Google App Engine for server-side data visualization. This framework has built a strong developer community enabling successful deployments worldwide. Sana's mobile platform also integrates with ODK [**?**]. Recent work [**?**] has explored creating a standard framework over Android's existing API for ODK to easily integrate with external sensors. ODK 2.0 provides an API for Bluetooth and USB to interface with external sensors, presumably easing integration of sensors within the data collection framework [**?**]. As part of our usability study in rural India, we focus on factors that affect usability of mobile-data collection systems with external Bluetooth-enabled sensors.

eMOCHA (electronic Mobile Open-source Comprehensive Health Application) is an ODK-based mobile data-collection system, developed by the Johns Hopkins Center for Clinical Global Health Education [**?**]. This platform connects medical care providers directly with patients using Android smartphones to enable data sharing in realtime. This framework provides better security by using a combination of 256-bit symmetric key and asymmetric key to encrypt all data stored on mobile devices. While this platform supports data collection using sensors embedded in an Android device such as GPS radios and cameras, it does not support communication with external medical sensors.

To summarize, none of the frameworks integrate provenance data that relates to a

medical reading and only ODK2.0 provides API to connect with external sensors.

Independent researchers and manufacturers of blood-pressure monitors alike have produced several patents that focus on the challenge of enforcing correct blood-pressure posture to increase reading accuracy. All the methods we encountered use additional hardware, such as inclination and pressure sensors [**?**, **?**, **?**]. In this thesis, we present a novel method to enable non-clinical workers to measure blood pressure correctly without the use of additional external sensors by leveraging the sensors embedded in common mobile tablet computers. We expect that with increasing access to tablet computers, particularly in the developing world, our method would be easier to integrate with existing mHealth deployments and would be more portable in the field than sensor-based solutions.

3 Implementation

The *mCollector* integrates four modules that provide:

1. a wireless sensor communication interface with the mCollectorApp over Bluetooth,
2. encryption of all recorded data,
3. log of all user interface events to assess app usability, and
4. a web interface to view recorded data.

We discuss the different modules in detail in the following sections.

3.1 mCollector design

We use external sensors to measure blood pressure and weight of screening participants. The mCollector also includes a portable printer, which the health operators can use to

Sensor	Accuracy	Power requirements	Data transmission
FORA D-15B BP monitor	± 2% of BP	4 AA batteries	RS232/Bluetooth
FORA D-30F BP monitor	± 2% of BP	4 AA batteries	RS232/Bluetooth
Tanita HD 351 weighing scale	± 3% of weight	4 AA batteries	Bluetooth/Zigbee
StarIO SM-S300 portable printer	N/A	2 AA batteries	Bluetooth/Zigbee

Table 1: mCollector sensors

print health reports to share with screened participants; see Table 1 for details. The medical sensors used in the system support Bluetooth 2.0. We chose Bluetooth for networking with medical sensors because most smartphones and tablets already support this technology. More importantly, once these sensors are paired with a mobile device, no additional configuration is required to transfer data between them. The Bluetooth standard's Health Device profile (HDP), which can be used to communicate with health devices such as heart-rate monitors, blood-pressure meters and weighing scales, is now supported on Android 4.0 (API level 14) [**?**]. At the time of our study, however, few manufacturers supported this profile in medical sensors. Therefore, we chose devices that support the more commonly used Serial Port Profile (SPP) on Bluetooth 2.0 that establishes an RFCOMM data channel and sets up virtual COM ports for applications to connect and send data [**?**]. Data from the sensors is streamed to the Android tablet over the RFCOMM channel, as shown in Figure 3.

3.2 Blood-pressure provenance

Blood pressure is difficult to measure accurately because it can be influenced by behavioral factors such as stress, smoking, caffeine intake, exercise, or natural factors like circadian variation. To compound these factors, incorrect techniques of measuring blood pressure can introduce additional errors. Our discussions with doctors in India and our understanding of the American Heart Association guidelines yield the following clinical

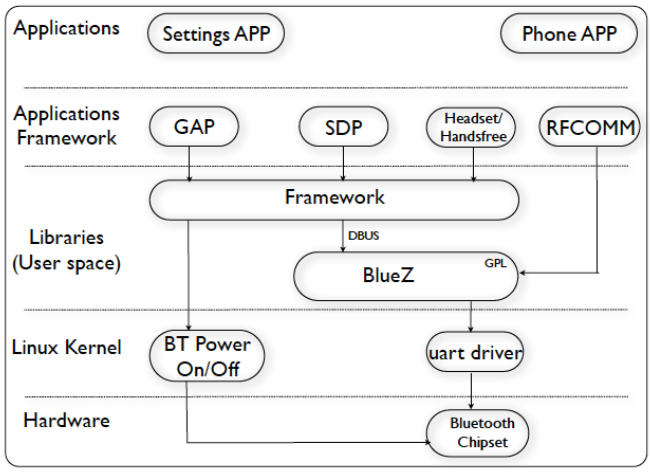

Figure 3: Android Bluetooth Architecture. *Source:*[**?**]

recommendation for blood-pressure measurement posture [**?**]:

1. The patient should be seated comfortably with the back supported and the upper arm bared without constrictive clothing.

2. The arm should be supported at heart level, and the bladder of the cuff should encircle at least 80% of the arm circumference.

3. Neither the patient nor the observer should talk during the measurement and the patient must not move her arm while the blood pressure is being taken.

Research has indicated that health workers are more prone to error in blood-pressure measurement than certified nurses, who are themselves more prone to error than physicians [**?**]. Therefore, it is useful to integrate a correcting component with the data-collection interface to guide non-clinical workers in realtime while they measure a participant's blood pressure.

Blood-pressure cuffs available in the market are already equipped with sensors to detect movement. These sensors are also programmed to repeat a reading when movement is detected because arm movements could perturb blood in the veins and thus result in an

erroneous reading. However, since the typical cuff and its embedded sensors attach to the upper arm, our tests show that lower-arm movement may go undetected. Lower-arm movement detection would also be particularly useful when wrist-worn blood-pressure monitors are used since wrist-worn monitors are more sensitive to posture than monitors that measure blood pressure in the upper arm [**?**].

3.3 Method

Our method can be used to verify two lower-arm posture requirements for blood-pressure measurements: first, it detects a subject's lower-arm movement, and second, it verifies whether the arm is well-supported on a flat surface for the duration of blood-pressure measurement. This method could enable non-clinical workers to measure blood pressure correctly without the use of dedicated sensors, unlike a previous pressure-sensor-based approach presented by Smithayer [**?**].

Regarding the latter challenge, we found that it was straightforward to use the tablet's accelerometer to determine whether the tablet remained well-supported throughout the BP measurement, by ensuring that the x and y (horizontal) accelerations each remained less than $1g$ and the z (vertical) acceleration remained in the range $7.1 - 8.5g$, where g is the Earth's acceleration due to gravity. Although we did not evaluate these thresholds in a field study, they appear to work well from initial experiments that we conducted on different surfaces and surface inclinations.

Thus, we focus on the challenge of determining whether the subject's lower arm remains still during a blood-pressure measurement, while it rests on the tablet screen. To begin, we describe how we collected a labeled data set for training a machine-learning model.

3.3.1 Data collection

We built a labeled data set for stationary and moving arms using an Android tablet's multi-touch touchscreen. We use the term *session* to refer to each 30-second subject session in which a set of touchscreen data is collected. A *moving session* is a session in which the subject was asked to move their lower arm, while it rested on the tablet, for 5-10 seconds. A *stationary session* is a session in which the lower arm remained motionless for the 30-second data collection interval. Thus, we create a labeled dataset of moving and stationary sessions.

For this experiment, we collected data from 12 subjects with each subject contributing five stationary and five moving sessions, for a total of 120 sessions in the dataset. The study procedures involving human subjects were approved by the institutional review board (IRB) at Dartmouth College. The data collection exercise was set up as follows:

1. The subject was instructed to rest her elbow on the tablet screen and subsequently lower her arm such that her lower arm was in contact with the touchscreen for 30 seconds, as shown in Figure 4.

2. For 'moving' sessions, the subject was asked to move her arm for approximately 5 seconds during the 30-second data collection interval. The classification would be trivial if a subject exaggerated her arm movements throughout the entire data collection duration. Indeed, initial experiments suggested that our classifier would detect small datasets limited to such exaggerated movements with 100% accuracy. This increased our confidence that linear classification was sufficient to classify moving arms from stationary ones. To capture realistic movements, the researcher demonstrated five examples of movement to the subject:

   (a) the lower arm slides along the x-axis,

   (b) the arm slides along the y-axis,

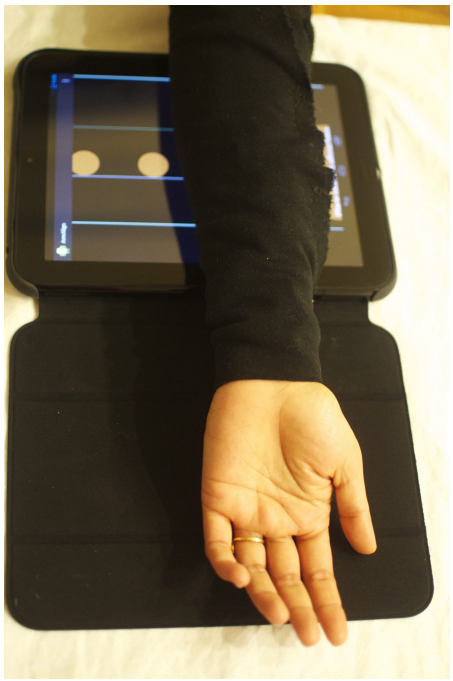

Figure 4: Using the tablet touchscreen to monitor lower-arm posture

(c) the arm slides along both x and y axes in either up or down direction,

(d) along any of the axes and between any two points on the touch screen, the arm slides in a sudden movement, and finally,

(e) along any of the axes and between any two points on the touch screen, the arm slides gradually from one point on the screen to another.

3. For 'stationary' sessions, the subject was instructed to keep her arm still after the arm was lowered onto the tablet.

4. While the subject rested her arm on the tablet touchscreen, our data-gathering app recorded the touchscreen activity throughout the session. It used standard Android APIs to record the set of *motion events* occurring during the measurement period.

During our preliminary experiments, when the entire skin surface of the lower arm came in direct contact with the tablet's capacitative touchscreen, we found that the

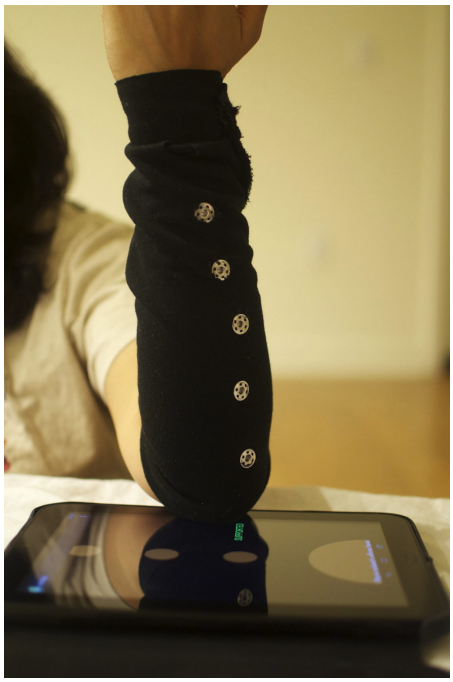

Figure 5: Protective cover to minimize skin contact with the touchscreen

touchscreen malfunctioned as a result of being overwhelmed by the sudden and high volume of static discharge. To address this problem, we designed a protective sleeve (with five conductive steel rings embedded) to localize skin contact with the touchscreen around the rings. The protective sleeve, shown in Figure 5, restricts direct skin contact with the touchscreen while allowing us to detect and track touch events that occur around the rings.

The Android framework defines a *motion event* to include the temporal history of one or more *touch events*. For example, a three-fingered swipe is a single motion event that describes three touch events, each describing the path and pressure of one finger as it swipes across the touchscreen.

To detect and track movements on the touchscreen, we describe each *touch event* as a series of four-tuples, each tuple including the timestamp t, the coordinates of the screen position (x, y), and the pressure of the touch p. These attributes capture *movement* in

sufficient detail and are commonly used in sample applications provided by the Android framework for gesture detection. In short, then, our approach is to analyze the changes to these four touch-event attributes, across all motion events within a session, to detect lower-arm movement in that session.

3.3.2 Algorithm

Given a set of labeled session data, as described above, we train a machine-learning algorithm to distinguish stationary sessions from moving sessions. First, we need an algorithm to generate a representative feature vector for each BP-measurement session using a subset of touch-event attributes.

Consider a session s. Let M be the set of motion events $m_1, \ldots m_i, \ldots m_n$ in this session.

For each m_i, there is a set of touch events m_{ij}.

For each m_{ij}, the app has recorded a sequence of attributes t, x, y, p; considered separately, these sequences are denoted t_{ij}, x_{ij}, y_{ij}, p_{ij}. More precisely, x_{ij} is a sequence $x_{ij1}, \ldots x_{ijk}, \ldots$, for some duration.

For each session s we build a matrix C where each row C_{ij} represents one event m_{ij}. Each such row is a 7-tuple:

$$C_{ij} = [i, j, \text{var}\{x_{ij}\}, \text{var}\{y_{ij}\}, \text{var}\{p_{ij}\}, X_{ij}, Y_{ij}]$$

where i, j serve to label the origin of the data in this row, where

$$\text{var}\{x_{ij}\} = \text{variance of } \forall_k\{x_{ijk}\}$$

and similarly for var$\{y_{ij}\}$, var$\{p_{ij}\}$, and where

$$X_{ij} = \text{range}\{x_{ij}\} = \max_k\{x_{ijk}\} - \min_k\{x_{ijk}\}$$

and similarly for Y_{ij}.

We summarize the C matrices by building the feature matrix F where each row represents one session s, as a 5-tuple that describes the maximum value along each dimension:

$$F_s = [\text{maxvar}\{x\}, \text{maxvar}\{y\}, \text{maxvar}\{p\}, \text{maxX}, \text{maxY}]$$

where maxvar is the maximum variance, that is,

$$\text{maxvar}\{x\} = \max_{ij}\{\text{var}\{x_{ij}\}\}$$

and similarly for maxvar$\{y\}$, maxvar$\{p\}$; and where

$$\text{maxX} = \max_{ij}\{X_{ij}\}$$

and similarly for maxY.

We then trained a binary SVM classifier using a subset of the features (columns) in F, and a subset of the labeled data collected earlier. Before we review the results of our experiments with this classifier, though, we need to explain how we chose a suitable subset of features for use in our classifier.

Given the above feature matrix F, we sought to determine which combination of features (columns of F) would yield a classifier with best performance. Figure 6 shows this *feature analysis*, comparing the accuracy of the SVM classifier with different feature

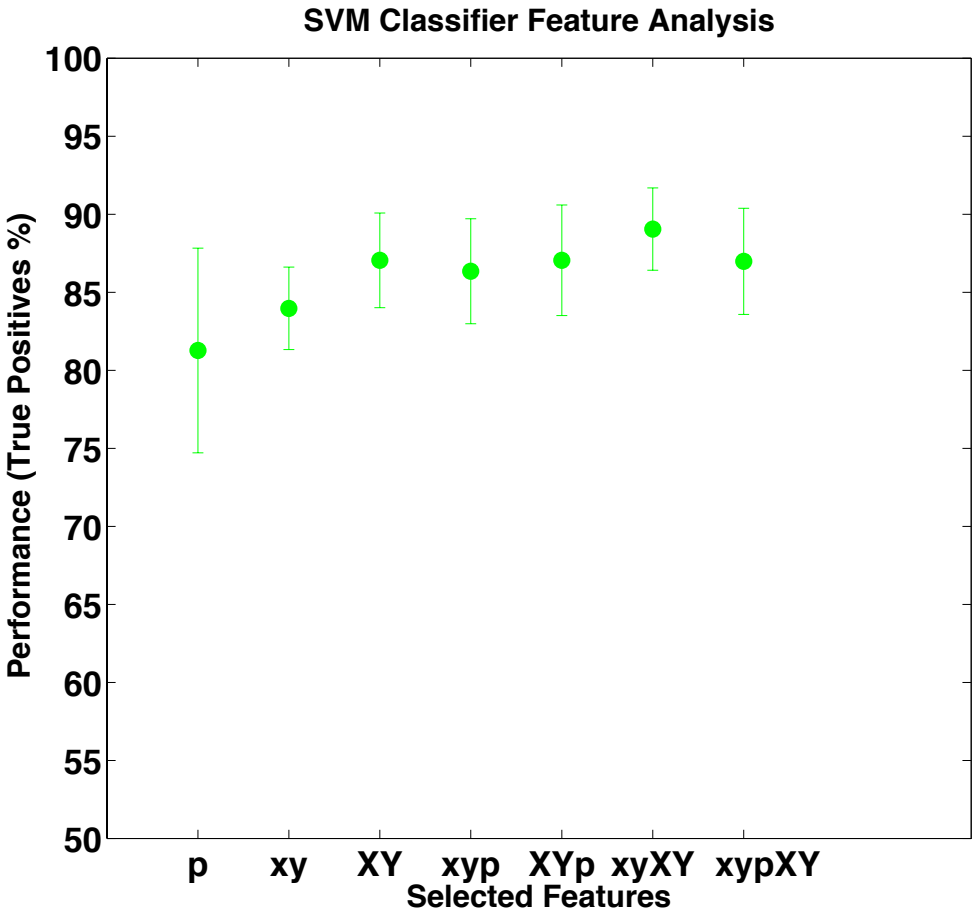

Figure 6: Feature analysis

sets. The features **x**, **y**, **p**, **X** and **Y** in Figure 6 correspond to the maxvar$\{x\}$, maxvar$\{y\}$, maxvar$\{p\}$, maxX and maxY columns of F respectively; hereon, we use this boldface notation to refer to these aggregate features. The features **X** and **Y** capture the largest movements along x and y axes in a given session. Note that the feature set consisting of **X** and **Y** would be insufficient to linearly separate moving and stationary sessions. Since universal thresholds such as absolute differences in x and y alone are not enough to distinguish stationary and moving arms reliably, we clearly need a classifier that captures nuances of movement on the touchscreen.

Although the **p** feature itself yields an adequate global performance accuracy of 81%, one can observe in this figure that the best feature set excludes the **p** feature. [A more extensive user study is required to understand whether diverse weight of subjects' arms might be expressed by **p** and potentially improve classification.] From these results, we chose the feature set [**x**, **y**, **X**, **Y**], which resulted in 91.3% accurate classification of all stationary and moving sessions in the test data.

We constructed F so moving sessions constituted the first half of the rows, and stationary sessions constituted the second half of the rows. We then selected sessions represented by odd-numbered rows from F as training data for the classifier, and selected sessions represented by even-numbered rows in F to constitute test data. Thus, the test and training partitions of the dataset matrix F never overlap. We use random subsets within these odd-even rows and average the classifier performance across those permutations to report the final results. The training partitions of F are input to a linear SVM classifier to learn classification weights for stationary and moving sessions. The classifier was applied to the test partition of F to measure accuracy.

4 Evaluation

In this chapter, we evaluate the performance of the classifier that detects lower-arm movement to collect BP provenance. We then present the results of the usability studies conducted on the mCollector.

4.1 BP Provenance method

Figure 7 and Table 2 show the results of our classifier, in which classifier accuracy is expressed as the rate of true positives.

Figure 7 shows the results for different movement types within the moving class

Table 2: Classifier results (True positives)

Category	Global	Moving	Stationary
All movements	91.3 %	86.8 %	95.8 %
Slow tremors	94.6 %	93.4 %	95.8 %
Swift movements	89.5 %	83.2 %	95.8 %

across all feature sets. The legends in Figure 7 indicate:

1. *Slow tremors:* the accuracy reported for the set of rows in the test dataset that correspond to slow tremor-like movements.

2. *Swift movements:* the accuracy reported for the set of rows in the test dataset that correspond to swift movements.

Upon examining the underlying data, we found that the range of variance for the values of the **x** and **y** features (that is, maxvar$\{x\}$ − minvar$\{x\}$) across small, tremor-like movements was less than the corresponding range for the same features for swift movements. This difference could explain why the linear classifier performed better with slow movements than with swift movements across most feature sets. From Figure 7, we observe that the addition of more features improves classification for swift movements, but less so for slow movements. This difference could be explained by the fact that the dataset contains more examples and variations of swift movements than stationary ones. We can conclude that the classifier can robustly detect different types of movement although the performance appears to be better for slow movements.

From the confusion matrix shown in Table 3, the false-negative rate for moving sessions is 15%; none of the actual stationary sessions were wrongly classified as moving.

Given the lack of standards and benchmarks to evaluate mHealth deployments [**?**], we present an evaluation plan in this section to measure the effectiveness of the mCollector deployment in rural regions. We propose *system usability* as a key metric to determine effectiveness of the mCollector.

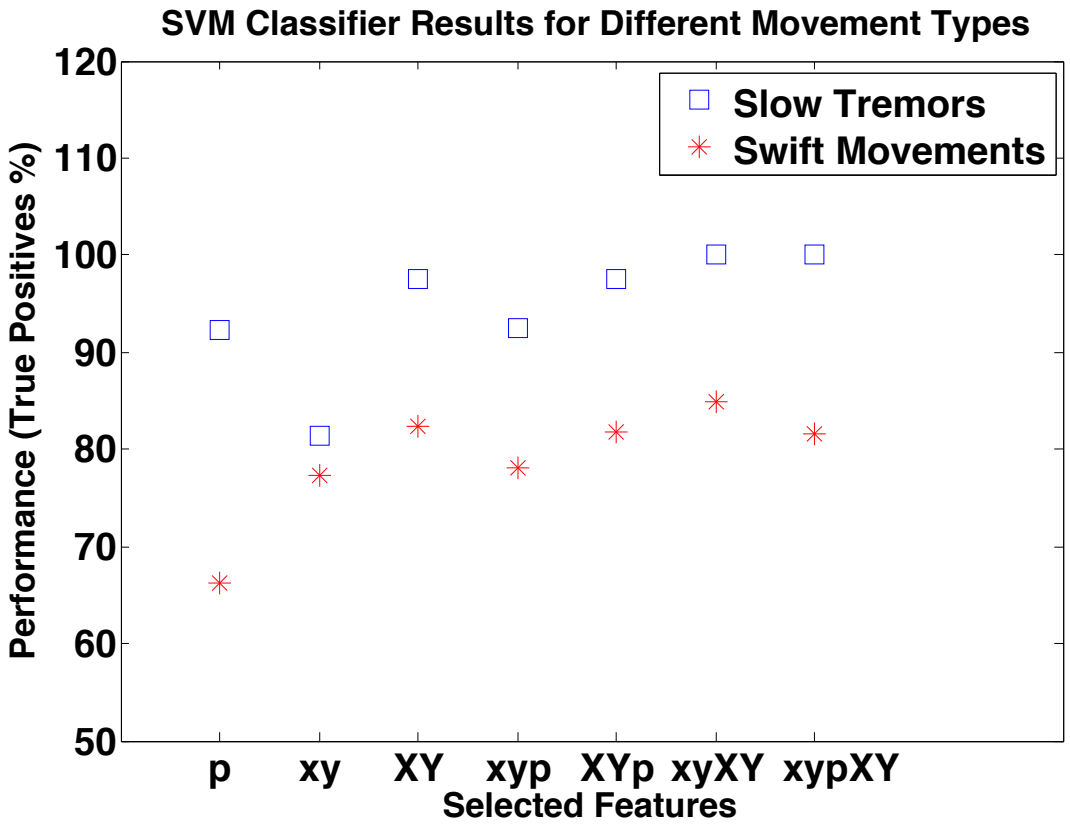

Figure 7: Prediction accuracy for moving sessions where the movement is known to be either slow tremors or swift movements. From the plot, the method is able to detect slow tremor-like movement better than swift movements.

Table 3: Confusion matrix

	Predicted	
Actual	Stationary	Moving
Stationary	1.0	0.0
Moving	0.15	0.84

4.2 mCollector usability study

To assess the mCollector's usability on the field, we integrated a mobile toolkit library that logged all app-user interaction such as swipes, button clicks and touchscreen taps. Initially, the developer's interaction with the mCollector app was recorded using the toolkit app. This usage data was used to create the mCollector's expected usage model, referred to as the *expert usage model*. Subsequent interaction of operators with the mCollector was recorded to create a *novice usage model*. The expert and novice usage models were compared and differences between the two models were analyzed to quantify app usability. All interface interaction logs were cached on the tablet's SD card until the tablet entered a wireless network; the logs were then transferred to a remote server. Services running on this remote server were used to generate various usability metrics to identify problems in the app's interface and quantify app usability. These usability metrics are discussed in the next section.

We conducted three user studies with our system. In the first two studies, the primary researcher used the mCollector herself to collect health data. In the first study, she collected health data of 19 graduate students. A second similar user study was conducted with 20 Dartmouth College employees. In the third study, we recruited three non-clinical, non-technical operators to use the mCollector app in the field at a rural mobile screening camp. The three operators collected health data from 26 participants at this screening camp in rural India using the mCollector. The first two pilot studies revealed some design issues in the app, which were fixed for the field study. However, the wireless connectivity of some of the sensors with the app remained unreliable and resulted in partial data loss in the rural field study.

Background. India is an information and communications technology (ICT) hub that hosted the highest number of rural mHealth projects in the world [**?**]. Therefore, we

chose India as a site to study factors that affect usability of our system. Although India has the second largest number of mobile phones in the world, there is a wide gap in telecommunication services between rural and urban regions in India; 72% of India's 1.21 billion population resides in rural regions. The mobile *teledensity*, defined as the number of cellular subscriptions for every 100 people, is only 35% for rural India in contrast with around 65% in urban India according to subscription data released by the Telecom Regulatory Authority of India (TRAI) [**?**]. Despite such disparities, and indeed because of them, recent ICT projects have focused on solving unique technological challenges in remote rural regions.

To observe existing systems that collect health data in remote locations, we accompanied our partners from the Narayana Hrudayalaya hospital's (NHH) mobile cardiac screening program on a cardiac screening event conducted in Kodhlapur village, located some 100 kms (130 miles) north of Bangalore, in January 2012. The mobile health camp included screening for hypertension, blood glucose, malnutrition and cardiac disease. Camp workers recorded participant registration information in handwritten notebooks but did not not aggregate or process this data further. Only patients identified as high risk for onset of cardiovascular disease were referred back to the NHH cardiac facility, in Bangalore, for diagnosis and subsidized treatment. However, we believe there is value in maintaining health records of the relatively healthy populations in remote, rural areas. For instance, medical data collected from different screening camp sites could be archived and compared with data collected in subsequent visits to track that population's health over time.[†]

[†]Note that unique identification of individual residents is still an open problem in most developing countries. India has begun to address this challenge via the Unique Identification Authority of India, which has proposed the *Aadhar* program as a solution to the permanent resident identification problem. This program is underway to create a national identity database and has registered 800 million Indians into the system as of April 2015. The system generates a unique 12-digit identification number as a function of a resident's biometrics so that services subsidized by the government such as health care, food, education, housing and guaranteed employment follow an individual and his/her family, as they migrate across villages and cities [**?**]. The potential to integrate mHealth programs with such a national identity database to deliver

Third User Study. To understand the challenges of deploying systems such as ours, we then conducted a field study in July 2013 in India with 24 rural participants and three app operators. Since the primary researcher/author of this work was unable to travel to India, Shrirang Mare (doctoral candidate, Dartmouth College) conducted the study on her behalf and recorded experiential challenges. He is referred to as the proxy researcher in this document.

We deployed the mCollector at NHH's mobile cardiac screening program conducted in a government-run hospital in Kolar district. We hired three operators from Bangalore to accompany us to the screening camp site. The mHealth data collection app was handed to these operators after a brief training session. They took turns to conduct the screening; an example is shown in Figure 8. They manually recorded participant profile data and medical history. The app also recorded the participant's arm posture to verify that the participant's lower arm remained stationary and was supported on a flat surface. On the last screen, the operator printed the medical data collected using a portable, wireless printer, and provided the printout to the participant. We recorded the entire screening process on video for further understanding deployment challenges in the field.

The usability toolkit described in the previous section logged all app operator interactions with the mCollector. We also interviewed the operators after each cardiac screening event to gather general system usability feedback. The demographic information we recorded about the operators indicated that none of the three operators had previously used tablets or had used any Android device. Although we lost participant posture data in the user study due to a glitch, the usability logs we did collect identified issues related to design of the provenance sleeve that was then incorporated to test the provenance component of the app.

---

healthcare and continuously monitor health trends reliably over time is exciting and is bound to be explored over the next few decades.

Figure 8: mCollector being used by an operator

4.2.1 Quantitative usability analysis

The logs collected by the toolkit were analyzed by Dr. Xiaoxiao Ma, of University of Massachusetts, Lowell, to derive key usability metrics of the mCollector app. The *task* of screening each patient using the mCollector app was divided into *stages* to track individual operator interactions with the app. The usability metrics listed below capture some of the challenges faced in the field and are discussed in this section.

1. Task completion time: Time taken by the operator to complete a single task, i.e., time taken to screen one patient.
2. Operator's task completion ratio: $\frac{\text{Number of tasks completed by operator}}{\text{Number of tasks attempted by operator}}$
3. Number of backtracking events at each stage: The number of times the "Back" button on the app screen was pressed at each stage in a given task. This metric could indicate confusion, for example, when the operator clicked on unexpected buttons or remained on the same screen for longer than the expert user.

The *user* in the usability plots refers to the novice user, in this case, the app operator. The operator completion ratio is 100%, which indicates that all three operators completed

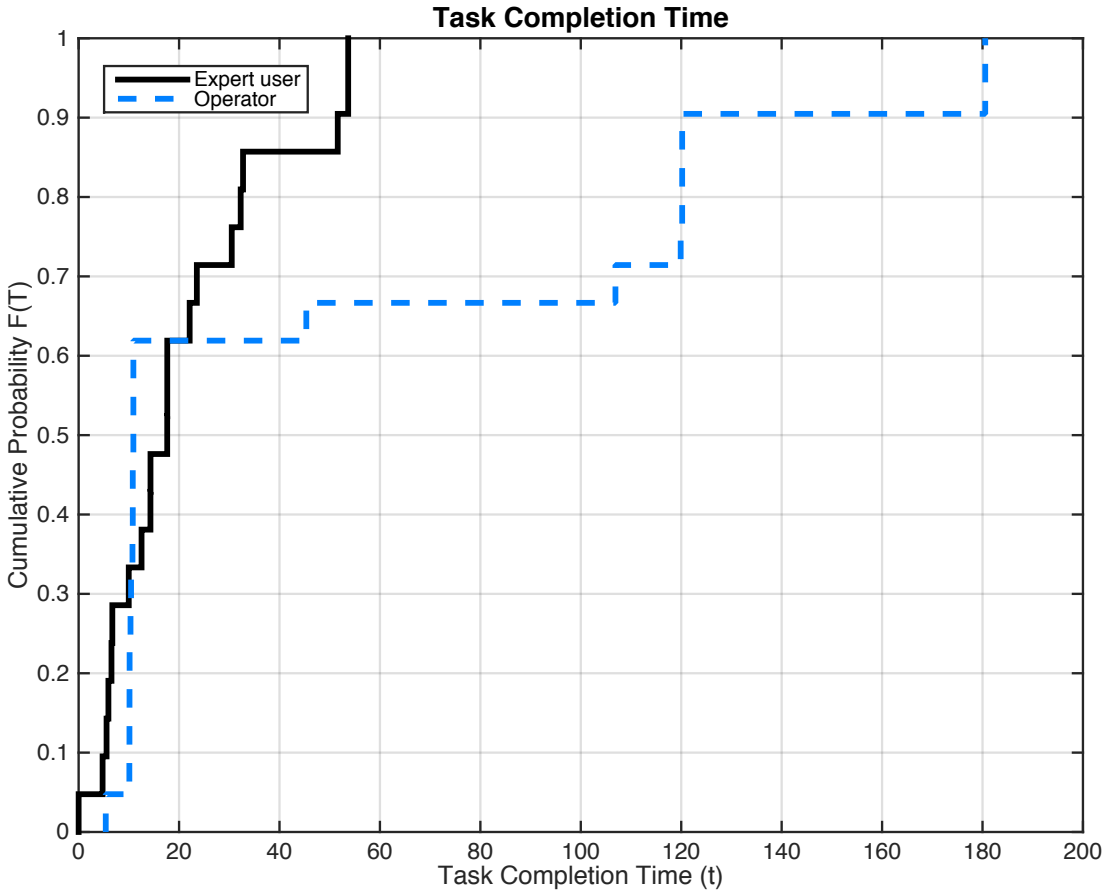

Figure 9: Usability results summary

screening of all participants. However, from Figure 9, the operator, on average, took nearly thrice as much time (183.08 seconds) as the app developer (53.68 seconds) to finish screening one participant. From the questionnaires we collected from the operators, we found that none of the operators had owned or used smartphones previously and, therefore, none of them were familiar with the Android platform. This factor could explain the wide gap in task completion time for operators and the app developer. It is also evident from the video recording of the screening that the operators were navigating the app screens very slowly for the first few participants. The mCollector app is a deck of screens which the operator steps through one at a time for data collection:

- Consent Checklist Screen: The first screen contains two questions with checkboxes to confirm that the patient is an adult who has consented to be part of this research.
- Profile Screen: The second screen collects a short medical history of the patient and does not record any personally identifiable information.
- Details Screen: The third app screen records height and weight.
- Provenance Screen: The fourth screen records the BP provenance using the patient's arm position.
- BP screen: The fifth screen connects to the wireless BP sensor to display a patient's BP reading.
- Printout Screen: Finally, the sixth screen shows a preview of all the collected data with the option of printing out the patient's data using a portable printer.

The different *stages* in the plots correspond to the following app screens.

- Stages 1-11: Consent checklist Screen
- 12-14: Details Screen
- 15: Provenance
- 16: BP Screen
- 17-20: SharePrintActivity

Figure 10 shows the time spent by operators across different screens of the app. The most confusing stages for all the operators were stages 12-20, which correspond to app screens that recorded blood-pressure-provenance, blood-pressure reading and the final print screen. Stages 12-14 corresponds to UI actions performed by the operator on the

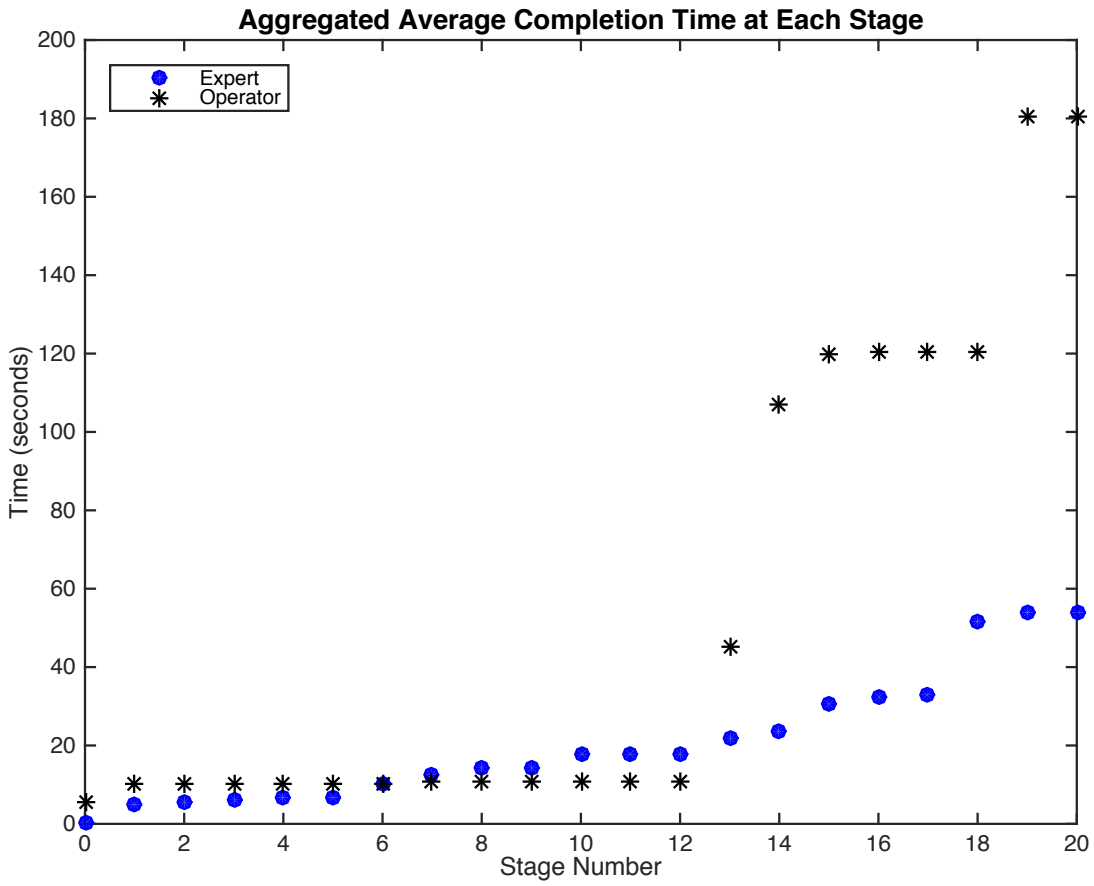

Figure 10: Aggregated average completion time

blood-pressure-provenance screen. Stages 15-18 correspond to the screen where the operator took a patient's BP reading.

Stages 19-20 correspond to the app screen where the operator printed a report of the patient data. Since we were unable to successfully pair the sensors with the tablet on the day of screening, the app's blood-pressure reading screen (which was previously automatic) had to be updated manually.

The tablet started to freeze when heavier arms were placed on the screen to collect blood-pressure provenance, which explains the sharp increase in time spent in stages 13-14. This also explains the steep increase in the number of backtracking events between

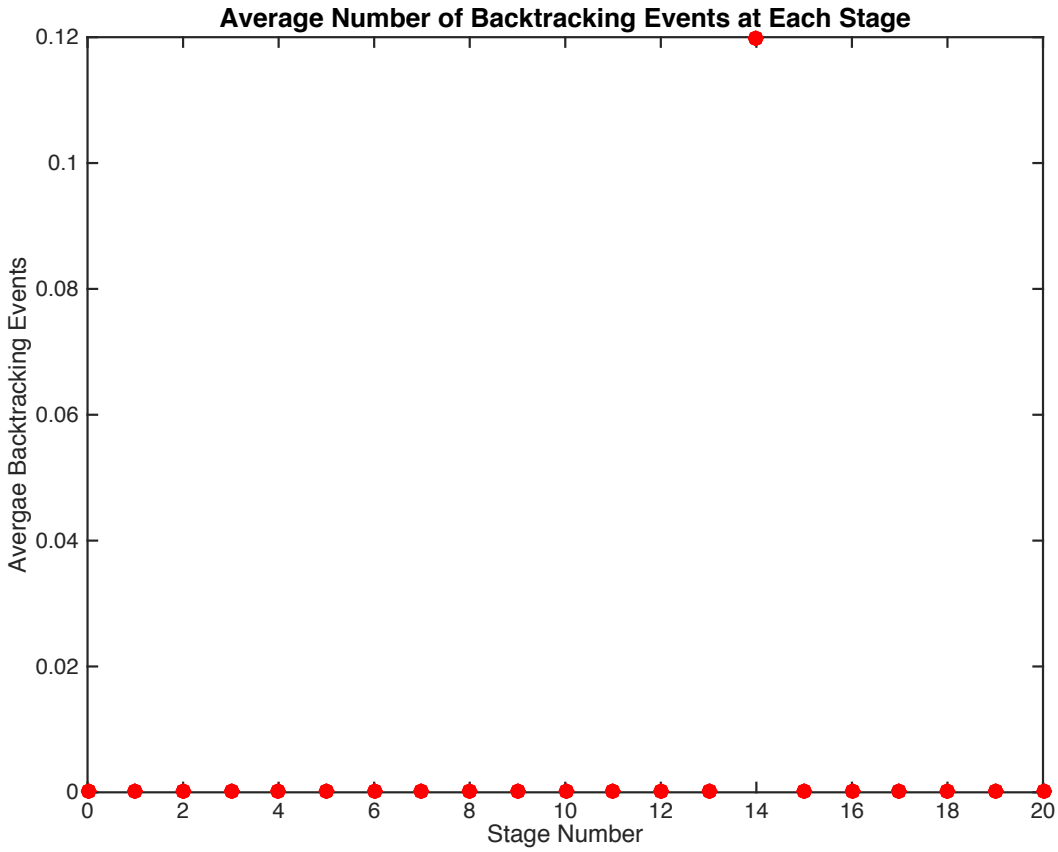

Figure 11: Average number of backtracking events at each stage

stages 13-15 shown in Figure 11, as the operators repeatedly tried to record data on the provenance screen each time the tablet froze, hitting the "Back" button to retry the operation.

4.2.2 Rural mHealth deployment: Lessons learned

Ease of use has been cited as an important barrier to mHealth adoption in India [**?**]. In this section, we compile a list of issues that caused unanticipated delays in our system pipeline which are likely useful to note for any rural mHealth deployment:

- *Entering patient's age*

  In India, birth dates prior to the 1960's may have not been recorded officially by

the state. Since some of the elderly patients did not remember or recall their age, they needed some time to recall or speculate.

- *Entering details on the first screen regarding the patient's condition (e.g., hypertension, breathing problem)*

  When the operator asked the subject about her condition, she would describe her symptoms the way she would to a doctor and the operator was not sure how to classify her problem among the conditions listed on the app screen. This would take time and often lead to incorrect data entry or no data entry. To speed up the user study, the operator was advised to skip entering these details.

- *Wireless sensors*

  We initially chose wireless sensors to make the system more portable and hassle-free. However, the last-minute challenges we faced with pairing and connecting the BP sensor with the tablet led us to conclude that it may be prudent to choose sensors that also work over wire, as a fallback, to avoid manual entry of data if wireless connections fail.

- *BP cuff attachment*

  The medically unskilled operators were taking more time than anticipated to attach the BP cuff to patients' arms. Since this was slowing data collection, the proxy researcher had one operator handle the tablet and do all the data entry, and another operator attached the BP cuff on the patient's arm to speed up the pipeline.

- *Taking the BP reading*

  If the operator did not set up the BP cuff correctly on the patient's arm, the BP monitor would take the second (sometimes third) reading automatically.

Due to delays in communicating with our partners at Narayana Hrudayalaya Hospitals in India, we only had a few hours to prepare for our deployment. We faced these technical challenges prior to deployment:

1. Bluetooth connectivity: The BP sensor would not connect to the tablet via Bluetooth; operators had to manually enter the BP readings.

2. Tablet firmware: While recording the touch events for BP provenance, the tablet would often hang due to pressure from the patient's arm, when the patient had heavy arms or exerted too much pressure.

3. Tablet SD card: Although we collected the usability data, the mCollector failed to record participant profile data due to permission errors on the tablet's SD card, and the app failed to report this error.

The pilot projects and health worker usability study procedures involving human subjects were approved by the Committee for Protection of Human Subjects, the institutional review board at Dartmouth College.

4.2.3 mCollector: Proposed changes

In this section, we propose changes to the mCollector design to address some of the challenges we faced in the field:

- *App display:* To ease data entry for operators, we should automatically adjust screen brightness from the mCollector app to adapt to ambient light. The operators may not be technically aware enough to choose Android's "Auto brightness" display setting.

- *Pre-pair sensors and tablets:* Any sensors, used to collect data in the field, must be pre-paired with the tablet before the deployment.

- *HCI (Human-Computer Interaction) guidelines:* Although we have not encountered a de facto standard for designing mobile interfaces for users in the developing world, HCI guidelines that recommend empathic design of mobile applications intended for rural users are available[**?**, **?**]. From our own experience in the field, we add some guidelines to this set:

    1. *Fail gracefully and have alternate workflows:* Using mHealth apps could be daunting to technically naive users. While it is important to fail gracefully in a general context, it is critical in the rural context. It is useful to have pre-defined workflows to handle different points of failure. In our study, when we failed to pair sensors with the tablet, we had to come up with a second workflow to enter data manually into the app in the field. As the usability study revealed in Figure 11, when the BP sensor failed to connect to the tablet, we had to work around this problem by having the operator navigate to the previous screen and repeat the BP measurement. We can redesign mCollector to allow the operator to repeat a reading on the same app screen.

    2. *Cultural sensitivity:* Some data fields in our app recorded birthdate and pre-existing health conditions. Some of our older patients could not recall their exact date of birth and reported an estimated age; the user interface design should be flexible enough to allow such entries to be marked accordingly to maintain data accuracy. And, although we clarified repeatedly that the volunteer operators of our app were not medical professionals, many patients described their physical symptoms in detail and the operators were unable to record this information since we had a simple symptoms checklist. In hindsight, a voice snippet or free-form text to record symptoms may be more suited to capture this information.

3. *App sustainability:* Developing mobile apps on open frameworks such as ODK Sensors would result in sustainable applications supported by mature developer communities.

- *Consent:* Since we were aware that our user study participants may not understand English, we carried duplicate copies of the consent form in the local language, Kannada. However, we were still required to explain the consent form to the participants as many of them were unable to read. It is useful to dedicate one person to manage the consent process.

5 Limitations

Medical sensors used in this study, which include Bluetooth radios, currently cost about 30-50% more than sensors that do not support this technology. However, we suspect that as the popularity of home health-monitoring devices grows in the west, the cost of Bluetooth-enabled sensors is likely to decrease over time.

During our interaction with doctors in India, we observed that most chronic-disease health monitoring and rural-screening projects focus on detection and diagnostics for late-stage patients rather than preventive care and continuous monitoring of population health in general. The cardiac screening camp we attended, for example, focused on the small subset of patients who were at high risk of cardiovascular disease and were likely to require surgery or advanced treatment. Doctors were present onsite to screen patients for cardiovascular diseases and only those patients recognized to be at high risk were scheduled to follow up with care providers. One month's supply of medicine to manage blood pressure was provided at no charge to some participants at the on-site doctor's discretion. Patients were expected to follow up with a doctor to adjust the medicine dosage after a month had elapsed. While non-medical patient data such as

name, birth date and place of birth was recorded manually into register notebooks, data of the relatively healthy population was not recorded at all. To track patient health longitudinally over time, data collection systems, such as ours, should leverage national identity databases, if available.

6 Conclusions and Future work

Any Android mHealth app could use the method presented here, to detect whether an individual's arm is stationary and supported throughout a blood-pressure reading. Furthermore, when posture recommendation are violated, the app could provide haptic feedback to alert the patient, by analyzing the movement data from the touchscreen in real time.

Indeed this method could be integrated easily into existing mHealth data collection apps built on the Android operating system because the classifier only requires aggregated attributes of the built-in events associated with touchscreen contact. However, since the classifier weights are derived entirely from tablet touchscreen data, adjustments to these weights may be required across different Android hardware.

Although the classifier performance at 91.3% may be sufficient to distinguish most stationary and moving arms, we could work with doctors to understand specific types of lower-arm movement that result in erroneous blood-pressure readings. The labels could then be refined so that the classifier recognizes the subset of lower-arm movements that are medically relevant.

We should conduct more user studies on a greater number and variety of subjects to analyze whether the classifier performance can be improved by tuning the SVM features. For example, a larger user study with a wide range of arm weights could reveal that

pressure at the point of touch on the screen can be used to distinguish moving and stationary arms. From the small user study we presented earlier, including pressure as an SVM feature did not improve classifier results. Further, varying touchscreen sensitivities across different tablets could also affect classifier performance. In our experiments we have used Motorola Xoom 3G and HP Touchpad tablets.

The sleeve used in our experiments could be redesigned to replace the steel rings with conductive fabric to increase patient comfort and allow for a larger area of skin contact on the touchscreen.